|  |  |  |
| --- | --- | --- |
|  | **Carátula para entrega de prácticas** | |
| Facultad de Ingeniería | | Laboratorio de docencia |

Laboratorios de computación

salas A y B

|  |  |
| --- | --- |
| *Profesor:* | Ing. Claudia Rodríguez Espino |
| *Asignatura:* | Fundamentos de Programación |
| *Grupo:* | 1102 |
| *No de Práctica(s):* | 10 |
| *Integrante(s):* | Alejandro Meneses Mercado |
|  |  |
|  |  |
| *Semestre:* | 2018-1 |
| *Fecha de entrega:* | 27 de octubre del 2017 |
| *Observaciones:* |  |
|  |  |

CALIFICACIÓN: \_\_\_\_\_\_\_\_\_\_

Práctica 10: Depuración de programas

Objetivo:

Aprender las técnicas básicas de depuración de programas en C para revisar de manera precisa el flujo de ejecución de un programa y el valor de las variables; en su caso, corregir posibles errores.

Desarrollo:

Primero revisamos lo que era la depuración de programas el cual es un ambiente que permite conocer exactamente el flujo de ejecución del programa, el valor que las variables adquieren, la pila de llamadas a funciones, entre otros aspectos.

Vimos que antes de depurar un programa se debe compilar el mismo.

También vimos algunas definiciones como la de error que es una acción humana que produce o genera un resultado incorrecto.

Otro concepto fue el de defecto o fault que es la manifestación de un error en el software y este defecto causa una falla la cual es una desviación del resultado esperado.

Vimos que la depuración es utíl cuando se desea optimizar el programa, es decir, realizar un análisis del mismo en ejecución para averiguar cuál es su flujo de operación y encontrar formas de mejorarlo por ejemplo, reducir el código, utilizar menos recursos llegando a los mismos resultados,etc.

También por si el programa tiene algún fallo, es decir, que el programa no muestra los resultados que se esperan para cierta entrada de datos debido a que se cometió algún error durante el proceso de diseño.

O porque el programa tiene un error de ejecución o defecto, que es cuando el programa está ejecutándose y éste se detiene inesperadamente.

Otra cosa que revisamos fueron las funciones que tienen los depuradores, las cuales son:

Que pueden ejecutar el programa, muestran el código fuente con el número de línea con el fin de emular la ejecución del programa sobre éste, se pueden poner puntos de ruptura para verificar hasta qué punto el programa se ejecuta sin problemas y en qué parte podría existir el error, se puede ejecutar la siguiente instrucción y la siguiente línea o también se puede ejecutar la línea o instrucción anterior y por último podemos visualizar el valor de las variables.

Trabjamos con la depuración de programas en DevC++, para eso tuvimos que tener el código escrito en C que querimos depurar y luego en la opción de Herramientas debiamos seleccionar la opción Menu del compildor y acceder a la pestaña Generación/Optimización de Código y finalmente, en la subpestaña Enlazador, activar la opción Generar Información de Depuración.

Finalmente aplicamos las opciones que nos da el depurador para depurar los programas.

Como actividad tuvimos que depurar 4 programas:

1.Programa que evalua una condicion y usa un contador para el resultado.

#include <stdio.h>

main()

{

int N, CONT, AS;

AS=0;

CONT=1;

printf("TECLEA UN NUMERO: ");

scanf("%i",&N);

while(CONT<=N)

{

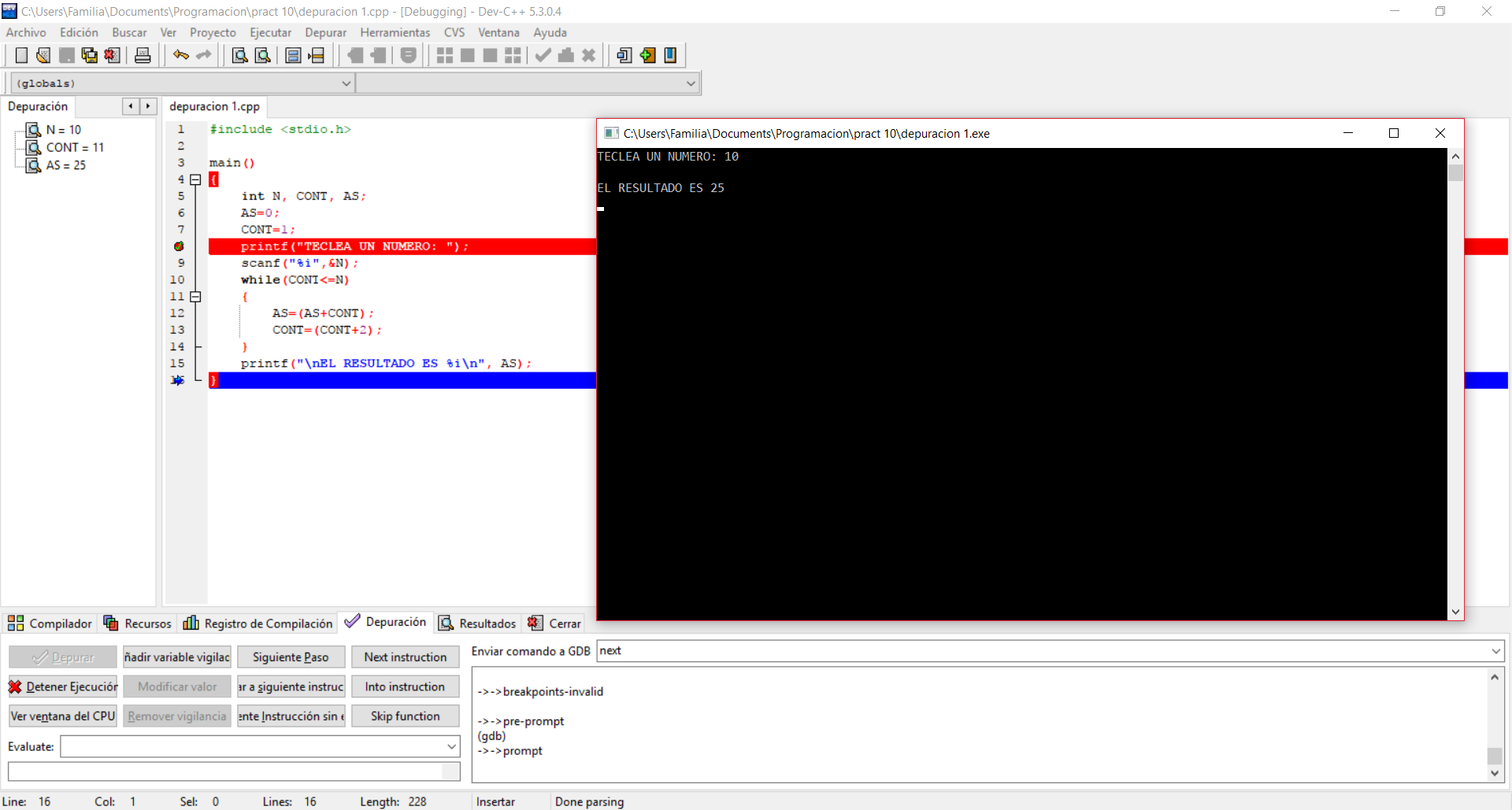
AS=(AS+CONT);

CONT=(CONT+2);

}

printf("\nEL RESULTADO ES %i\n", AS);

}



2.Programa que muestra las tablas de multiplicar del 1 al 10

#include <stdio.h>

main()

{

int i, j;

for(i=1; i<=10; i++)

{

printf("\nTabla del %i\n", i);

for(j=1; j<=10; j++)

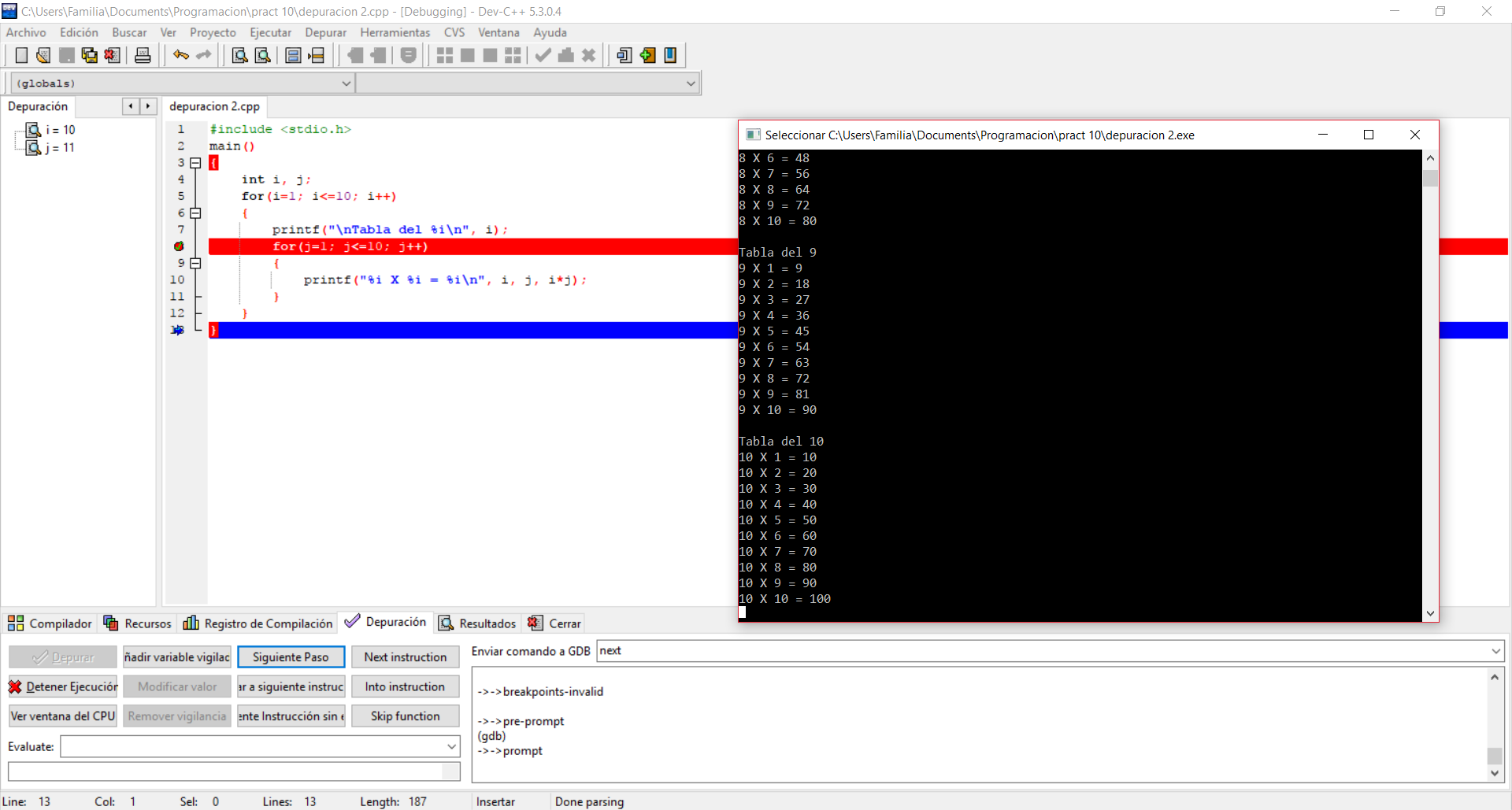
{

printf("%i X %i = %i\n", i, j, i\*j);

}

}

}



3. Programa que muestra el termino generico de la serie: X^K/K!

#include <stdio.h>

#include <math.h>

main()

{

int K, X, N;

float AS,AP;

printf("EL TERMINO GENERICO DE LA SERIE ES: X^K/K!");

printf("\nN=");

scanf("%d",&N);

printf("X=");

scanf("%d",&X);

K=0;

AP=1;

AS=0;

while(K<=N)

{

AS=(AS)+(pow(X,K));

AS=AS/AP;

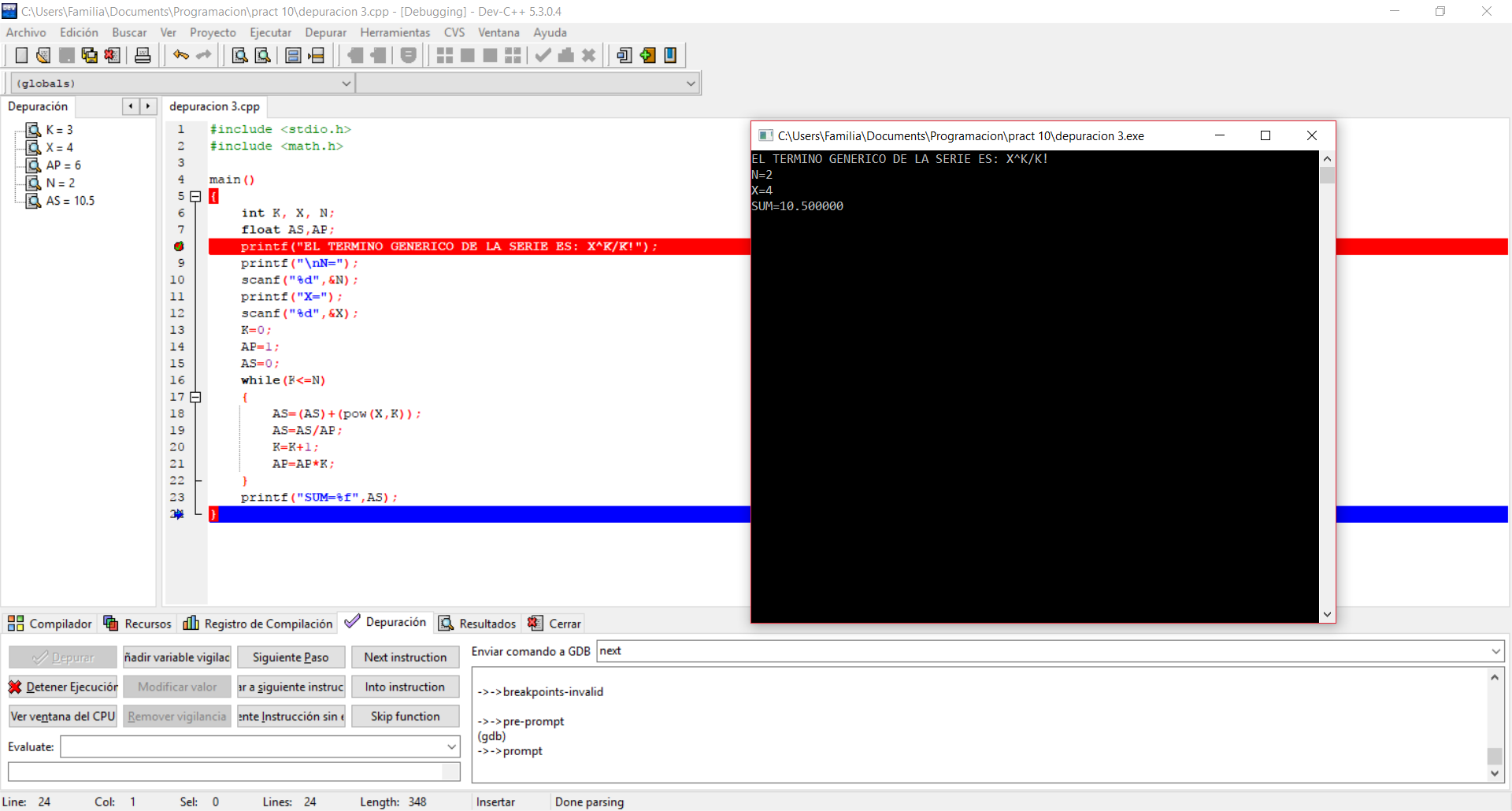
K=K+1;

AP=AP\*K;

}

printf("SUM=%f",AS);

}



4. Programa que resuleve ecuaciones con formula general.

#include<stdio.h>

#include<math.h>

int a,b,c,d;

float x1,x2,z,z1,r,i;

main()

{

printf("ingresa el valor de a: ");

scanf("%d",&a);

printf("\nIngresa el valor de b: ");

scanf("%d",&b);

printf("\nIngresa el valor de c: ");

scanf("%d",&c);

if(a==0)

{

printf("\nEl valor de a debe ser distinto de cero");

}

else

{

d=(b\*b)-(4\*a\*c);

if(d<0)

{

z=d\*(-1);

z1=sqrt(z);

r= -(b)/(2\*a);

i=(z1)/(2\*a);

printf("\nx1= %.2f + %fi", r,i);

printf("\nx2= %.2f - %fi",r,i);

}

else

{

x1=((-1\*b)+(sqrt(d)))/(2\*a);

x2=((-1\*b)-(sqrt(d)))/(2\*a);

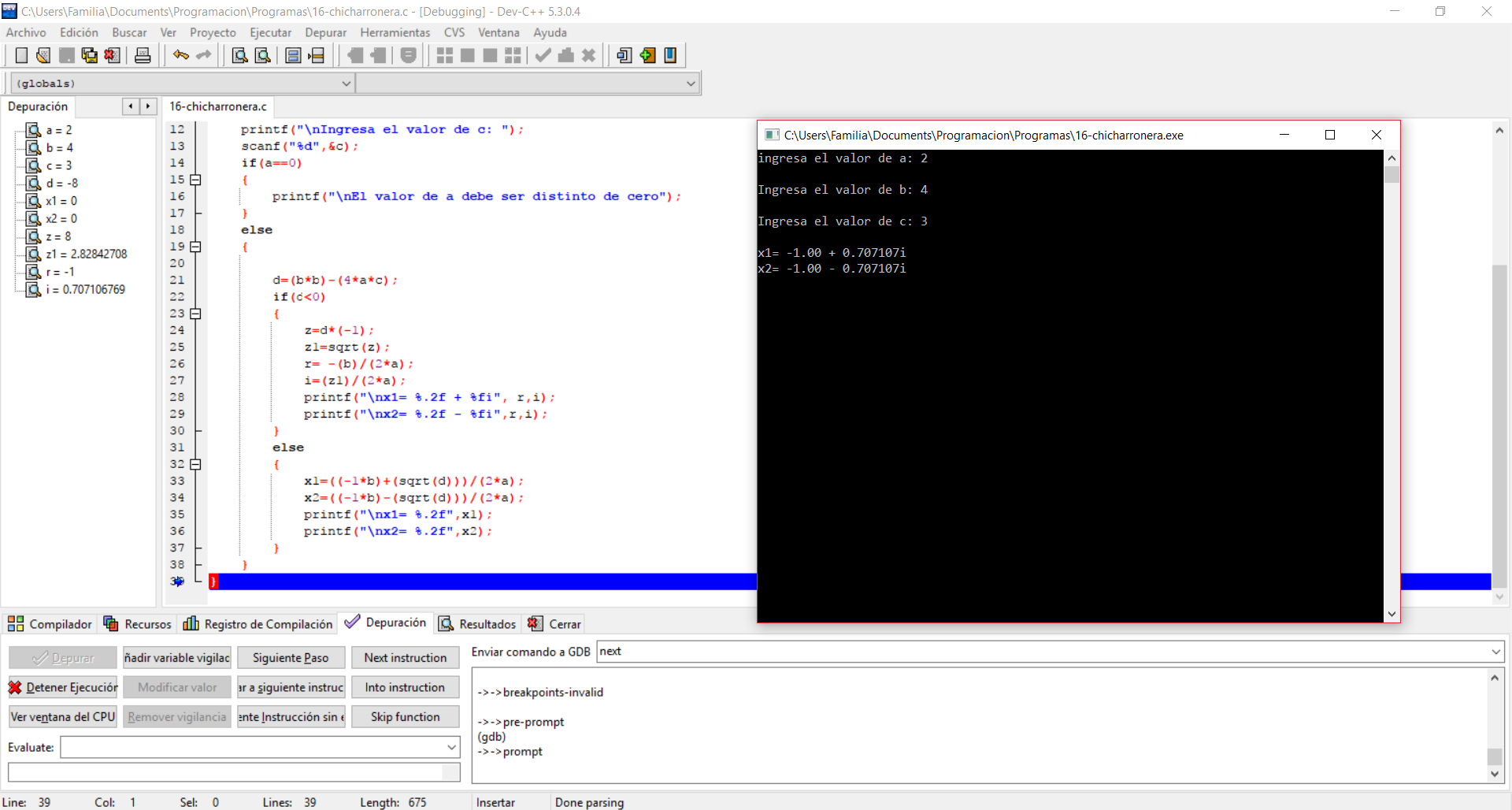
printf("\nx1= %.2f",x1);

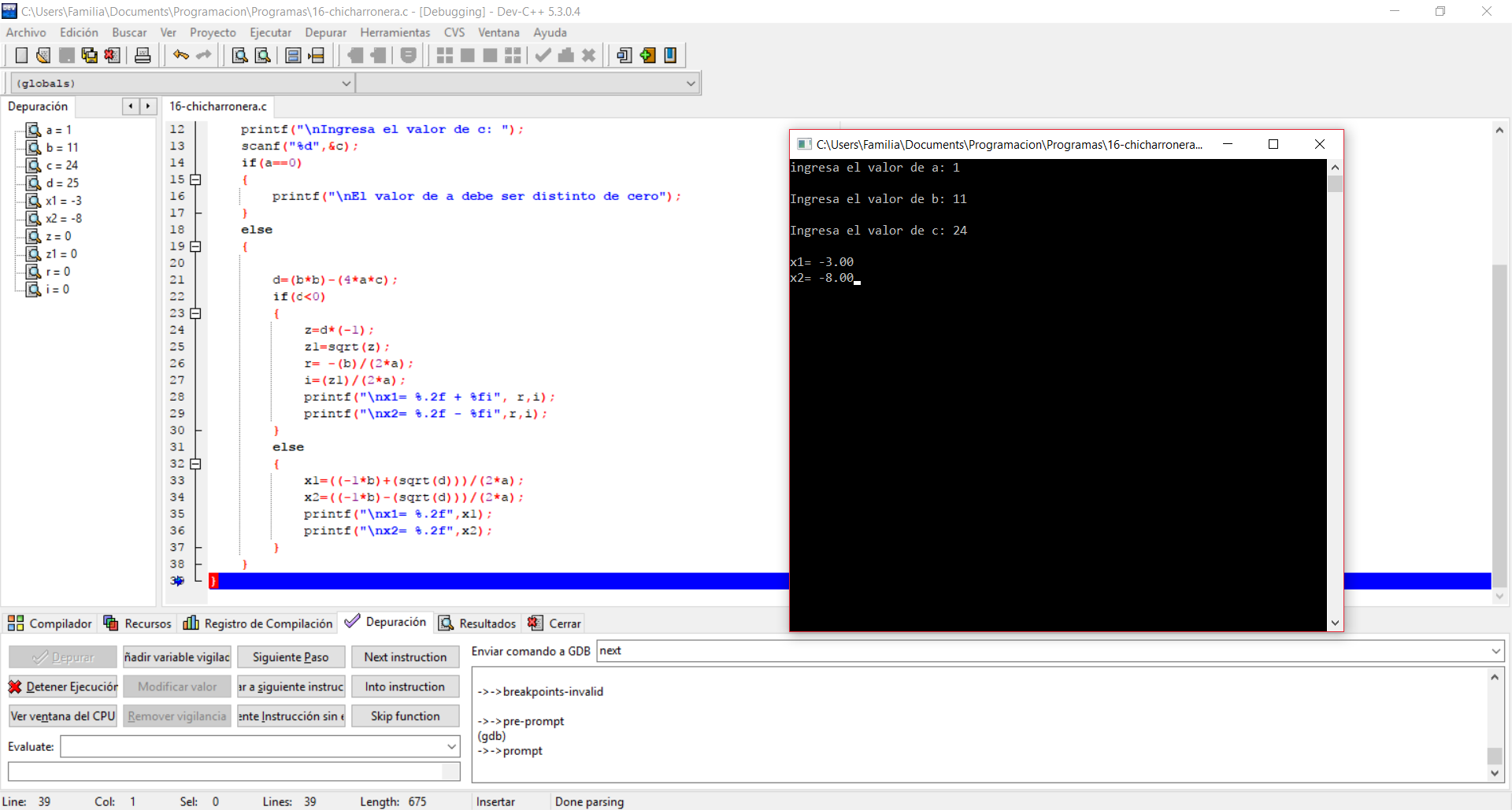
printf("\nx2= %.2f",x2);

}

}

}





Conclusiones:

Concluimos que gracias a esta práctica entendimos el funcionamiento de un depurador y que nos ayudara mucho en el futuro para poder optimizar nuestros programas o encontrar fallos en este.